

Experiment No : 2

Basic Statistical Analysis

2.1 Objective

Calculate and interpret the mean, median, mode, standard deviation, minimum, maximum and range of numerical student attributes. Use boxplots to inspect variation and possible outliers.

Important: The dataset is still raw. Missing values, invalid percentages and unusual observations are intentionally retained. Calculate the statistics, but interpret them cautiously.

2.2 Learning outcomes

After completing this experiment, you should be able to:

1. select numerical columns from a DataFrame;
2. calculate measures of central tendency
3. calculate measures of spread
4. compare the mean and median
5. identify suspicious minimum and maximum values; and
6. use boxplots to locate observations requiring investigation.

2.3 Theory

- i) **Measures of Center (Typicality):** These describe the "middle" or most typical value of a dataset. The experiment highlights three distinct measures:

Mean: The mean is the arithmetic average. Pandas adds the available values in each column and divides by the number of non-missing values. The mean uses every available value, so invalid percentages and extreme observations can influence it strongly.

Median: The median is the middle value after the observations are arranged in order. It is less sensitive to unusually high or low observations than the mean.

Mode: The mode is the most frequently occurring value. A column can have more than one mode.

- ii) **Measures of Spread (Variation):** These describe how dispersed or clustered the data points are around the center.

Standard Deviation: The standard deviation measures how far values typically lie from the mean. A small standard deviation indicates greater consistency; a large standard deviation indicates greater variation.

Range: The range is the maximum minus the minimum. It measures the complete observed span, but it is highly sensitive to extreme or invalid values.

- iii) **Measures of Boundaries (Extremes):**

Minimum and Maximum: The minimum is the smallest observed value and the maximum is the largest. These values are important data-quality checks. For percentage columns, values outside 0 to 100 are logically invalid.

- iv) **Boxplots:** A boxplot displays the median, quartiles, spread and possible outliers. One boxplot is drawn for each numerical column so variables with different measurement scales remain readable.

2.4 Program

Import the required libraries - Pandas for data analysis and pyplot for creating graphs

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

```
# Load the student CSV dataset
```

```
df = pd.read_csv("Student Academic Performance Record.csv")
```

Select the numerical attributes

Statistical measures such as the mean and standard deviation require numerical data. Identifiers, student names and category labels are therefore excluded.

List the numerical columns used in this experiment.

```
numeric_columns = [  
    "PU_Marks_Pct",  
    "Attendance_Pct",  
    "Engineering_Marks_Pct",  
    "Internships_Completed",  
    "Hackathons_Participated",  
    "Projects_Completed",  
    "Aptitude_Test_Pct",  
    "Placement_Package_LPA"  
]
```

Create a DataFrame containing only those columns.

```
numeric_data = df[numeric_columns]  
numeric_data.head()
```

	PU_Marks_Pct	Attendance_Pct	Engineering_Marks_Pct	Internships_Completed	Hackathons_Participated	Projects_Completed	Aptitude_Test_Pct
0	84.6	72.0	74.0	0.0	1.0	3	76.3
1	86.3	86.2	71.7	1.0	2.0	2	69.0
2	78.1	86.4	77.2	2.0	2.0	2	71.4
3	73.6	82.6	82.1	2.0	3.0	2	65.3
4	74.3	78.7	70.1	2.0	1.0	1	59.9

Calculate the arithmetic mean of every selected column.

```
numeric_data.mean().round(2)
```

PU_Marks_Pct	77.68
Attendance_Pct	83.01
Engineering_Marks_Pct	74.88
Internships_Completed	1.61
Hackathons_Participated	2.03
Projects_Completed	1.70
Aptitude_Test_Pct	68.62
Placement_Package_LPA	5.39

dtype: float64

Calculate the median of every selected column.

```
numeric_data.median().round(2)
```

PU_Marks_Pct	77.3
Attendance_Pct	82.8
Engineering_Marks_Pct	74.1
Internships_Completed	1.0
Hackathons_Participated	2.0
Projects_Completed	2.0
Aptitude_Test_Pct	68.4
Placement_Package_LPA	5.3

dtype: float64

Find the modes and display the first mode for each column.

```
numeric_data.mode().iloc[0].round(2)
```

PU_Marks_Pct	101.0
Attendance_Pct	105.0
Engineering_Marks_Pct	112.0
Internships_Completed	1.0
Hackathons_Participated	1.0
Projects_Completed	1.0
Aptitude_Test_Pct	69.8
Placement_Package_LPA	5.3

Name: 0, dtype: float64

Calculate the sample standard deviation.

```
numeric_data.std().round(2)
```

PU_Marks_Pct	9.01
Attendance_Pct	6.61
Engineering_Marks_Pct	8.93
Internships_Completed	1.47
Hackathons_Participated	2.05
Projects_Completed	1.29
Aptitude_Test_Pct	7.60
Placement_Package_LPA	0.77

dtype: float64

Display the smallest value in every selected column.

```
numeric_data.min()
```

PU_Marks_Pct	55.0
Attendance_Pct	65.0
Engineering_Marks_Pct	51.8
Internships_Completed	0.0
Hackathons_Participated	0.0
Projects_Completed	0.0
Aptitude_Test_Pct	47.2
Placement_Package_LPA	3.5

dtype: float64

Display the largest value in every selected column.

```
numeric_data.max()
```

PU_Marks_Pct	101.0
Attendance_Pct	105.0
Engineering_Marks_Pct	112.0
Internships_Completed	8.0
Hackathons_Participated	12.0
Projects_Completed	5.0
Aptitude_Test_Pct	92.8
Placement_Package_LPA	8.0

dtype: float64

```
# Calculate the range of every selected column.
```

```
numeric_data.max() - numeric_data.min()
```

```
PU_Marks_Pct          46.0
Attendance_Pct         40.0
Engineering_Marks_Pct  60.2
Internships_Completed   8.0
Hackathons_Participated 12.0
Projects_Completed      5.0
Aptitude_Test_Pct      45.6
Placement_Package_LPA   4.5
dtype: float64
```

```
# Combine all measures into one summary table.
```

```
statistics = pd.DataFrame({
    "Mean": numeric_data.mean(),
    "Median": numeric_data.median(),
    "Mode": numeric_data.mode().iloc[0],
    "Standard Deviation": numeric_data.std(),
    "Minimum": numeric_data.min(),
    "Maximum": numeric_data.max(),
    "Range": numeric_data.max() - numeric_data.min()
})
```

```
# Round the displayed values to two decimal places.
```

```
statistics.round(2)
```

	Mean	Median	Mode	Standard Deviation	Minimum	Maximum	Range
PU_Marks_Pct	77.68	77.3	101.0	9.01	55.0	101.0	46.0
Attendance_Pct	83.01	82.8	105.0	6.61	65.0	105.0	40.0
Engineering_Marks_Pct	74.88	74.1	112.0	8.93	51.8	112.0	60.2
Internships_Completed	1.61	1.0	1.0	1.47	0.0	8.0	8.0
Hackathons_Participated	2.03	2.0	1.0	2.05	0.0	12.0	12.0
Projects_Completed	1.70	2.0	1.0	1.29	0.0	5.0	5.0
Aptitude_Test_Pct	68.62	68.4	69.8	7.60	47.2	92.8	45.6
Placement_Package_LPA	5.39	5.3	5.3	0.77	3.5	8.0	4.5

```
# Subtract the median from the mean and sort the differences.
mean_median_difference = numeric_data.mean() - numeric_data.median()
mean_median_difference.sort_values(ascending=False).round(2)
```

```
Engineering_Marks_Pct      0.78
Internships_Completed      0.61
PU_Marks_Pct               0.38
Aptitude_Test_Pct          0.22
Attendance_Pct              0.21
Placement_Package_LPA      0.09
Hackathons_Participated    0.03
Projects_Completed         -0.30
dtype: float64
```

```
# Create a separate horizontal boxplot for every numerical column.
```

```
for column in numeric_columns:
```

```
    plt.figure(figsize=(8, 3))
```

```
    plt.boxplot(numeric_data[column].dropna(), vert=False)
```

```
    plt.title(f'Boxplot of {column}')
```

```
    plt.xlabel(column)
```

```
    plt.show()
```





